What Is Claimed Is:

1. A method for facilitating typesafe software design while supporting structured composition of a software system, comprising:

receiving a first invocation of the software system;

assigning a first context to the first invocation;

examining the first invocation to locate components of the first invocation;

registering a unique factory to build each component, wherein these factories are registered using the first context; and

when a component is needed, building the component using a factory associated with the component, whereby building the component after each component has a registered factory eliminates potential problems with initialization circularity.

2. The method of claim 1, further comprising:

receiving a second invocation of the software system;

assigning a second context to the second invocation;

examining the second invocation to locate components of the second invocation;

registering a unique factory to build each component, wherein these factories are registered using the second context; and

when a component is needed, building the component using a factory associated with the component, whereby building the component after each component has a registered factory eliminates problems with initialization circularity.

3. The method of claim 2, wherein components from the second invocation are not available to the first invocation.

4. The method of claim 1, further comprising providing an additional factory for an extended component of the first invocation.

5. The method of claim 1, wherein registering the unique factory to build each component involves placing a key and a related factory identifier into a storage structure.

6. The method of claim 5, wherein building the component using the factory associated with the component involves using the key to retrieve the related factory identifier from the storage structure.

7. The method of claim 6, wherein the storage structure includes a hash table.

8. A computer-readable storage medium storing instructions that when executed by a computer cause the computer to perform a method for facilitating typesafe software design while supporting structured composition of a software system, the method comprising:

receiving a first invocation of the software system;

assigning a first context to the first invocation;

examining the first invocation to locate components of the first invocation;

registering a unique factory to build each component, wherein these factories are registered using the first context; and

when a component is needed, building the component using a factory associated with the component, whereby building the component after each component has a registered factory eliminates potential problems with initialization circularity.

9. The computer-readable storage medium of claim 8, the method further comprising:

receiving a second invocation of the software system;

assigning a second context to the second invocation;

examining the second invocation to locate components of the second invocation;

registering a unique factory to build each component, wherein these factories are registered using the second context; and

when a component is needed, building the component using a factory associated with the component, whereby building the component after each component has a registered factory eliminates problems with initialization circularity.

10. The computer-readable storage medium of claim 9, wherein components from the second invocation are not available to the first invocation.

11. The computer-readable storage medium of claim 8, the method further comprising providing an additional factory for an extended component of the first invocation.

12. The computer-readable storage medium of claim 8, wherein registering the unique factory to build each component involves placing a key and a related factory identifier into a storage structure.

13. The computer-readable storage medium of claim 12, wherein building the component using the factory associated with the component involves using the key to retrieve the related factory identifier from the storage structure.

14. The computer-readable storage medium of claim 13, wherein the storage structure includes a hash table.

15. An apparatus for facilitating typesafe software design while supporting structured composition of a software system, comprising:

a receiving mechanism configured to receive a first invocation of the software system;

an assigning mechanism configured to assign a first context to the first invocation;

an examining mechanism configured to examine the first invocation to locate components of the first invocation;

a registering mechanism configured to register a unique factory to build each component, wherein these factories are registered using the first context; and

a building mechanism configured to build the component using a factory associated with the component when a component is needed, whereby building the component after each component has a registered factory eliminates potential problems with initialization circularity.

16. The apparatus of claim 15, wherein:

the receiving mechanism is further configured to receive a second invocation of the software system;

the assigning mechanism is further configured to assign a second context to the second invocation;

the examining mechanism is further configured to examine the second invocation to locate components of the second invocation;

the registering mechanism is further configured to register a unique factory to build each component, wherein these factories are registered using the second context; and

the building mechanism is further configured to build the component using a factory associated with the component when a component is needed, whereby building the component after each component has a registered factory eliminates problems with initialization circularity.

17. The apparatus of claim 16, wherein components from the second invocation are not available to the first invocation.

18. The apparatus of claim 15, further comprising a providing mechanism configured to provide an additional factory for an extended component of the first invocation.

19. The apparatus of claim 15, wherein registering the unique factory to build each component involves placing a key and a related factory identifier into a storage structure.

20. The apparatus of claim 19, wherein building the component using the factory associated with the component involves using the key to retrieve the related factory identifier from the storage structure.

21. The apparatus of claim 20, wherein the storage structure includes a hash table.